EXHIBIT B

In the Matter of:

Berkeley J. Bates Seminar

Youtube Video

---

68 Commercial Wharf • Boston, MA 02110
888.825.3376 - 617.399.0130
Global Coverage
court-reporting.com

O'BRIEN & LEVINE
COURT REPORTING SOLUTIONS

Google Exhibit 1022
Google v. Singular

COMMONWEALTH OF MASSACHUSETTS

YOUTUBE VIDEO

ON

PRACTICAL APPROXIMATE COMPUTING

AT

BERKELEY UNIVERSITY OF CALIFORNIA

SPEAKER: JOSEPH BATES

DATE: MARCH, 2016

MARY INDOMENICO, ACT, CET

Official Court Transcriber

MR. BATES: I'm a computer scientist with a -- I don't know -- sore throat or something, so I hope you can hear me.

I am currently at a start-up, small company. Used to be at Carnegie Melon AI faculty, and MIT, in certain roles. And I want to talk about this approximate computing idea, and what I think is a practical simple approach to it.

So, I about ten years ago belatedly realized what Carver Mead had been saying for a long time, which is that wires -- you know, junctions can add currents, and transistors can do exp and log. These are useful operations in a lot of tasks. And a modern chip -- you know, the chips in our cell phones, right, have whatever billion transistors. They run whatever gigahertz. So, in this simple sense of operation -- I mean they're running digitally, but you know, they have the capacity to do something like ten to the eighteenth ops per second. That's probably more than at least my brain is doing, best I understand neuroscience at least at a -- you know, a certain level of abstraction.

But as a programmer, we only get essentially zero of it. You know, its 0.0000, some more zeros percent. One out of a billion, maybe; ten out of a

billion. And I'm an AI guy. I'm getting to be an older AI guy. And I'd like to see my childhood dreams come true before, you know, before its over for me.

So, one theory about what's made AI successful to the extent it has been or it hasn't been, has been the material scientists and physicists; no Einstein of AI so far. Forgive me to any of my AI colleagues who may be here. And so if that's true -- and deep learnings, of course an example, right. I mean known that in the 60s, known that's in the 80s, known that's in the rest -- five years especially.

And so I, at least, think we or I desperately need to see that compute in the world. I don't want it going to make sure Windows runs in the next generation of laptops.

So, about a decade ago, I began to wonder about this. And Carver said, you know, there's a way you can do this. Look, just become a EE, not a computer scientist, become a EE, but be a CMOS EE, and make sure it's analoged and do it in sub threshold.

Thank you. Thanks a lot.

That turns out to be hard -- at least I found it to be hard. I tried to do it. In fact, Jacob helped me try to do it. But it seemed like there were

hundreds of people in the world skilled at this, and I wanted there to be hundreds of thousands of people. So, I asked myself the question: Could I take his ideas and try to make a quote "normal" computer? And si this is just going to tell you about a normal computer built -- sort of inspired by these ideas.

So, suppose you look at machines that did arithmetic that was pretty close. And the reason for this is that, you know, multiply and divide circuits are very large in silicon. So, I said look, you know, I'm going to make life easier for the hardware people and possibly somewhere between more difficult and impossible for the software people. What would happen? And so the spec I wrote down was not the I EEE 754 floating point spec, you know, like that. It was that. That's the spec. People ask me: "Well what's the distribution of the errors?" Don't think about it. Don't program to it. Don't try to figure it out. Okay? And the reason is going to be, you say this, the hardware people get a lot of flexibility to do crazy stuff to make tiny circuits. And you want tiny circuits if you're in certain fields of science, 'cause you want a lot of compute.

Now, the questions would be, you know: What

happens to the hardware? What happens to the software? Is it impossible to program with a machine that operates like that?

So, that's what I want to tell you about -- some of the results that I was able to find.

So, again, a lonely programmer was able -- me, was able to take a floating point unit -- traditional digital floating point unit and shrink it about 100X. Now that's potentially very good -- potentially very good. And we'll look at the issues. This was using conventional digital silicon. And again, Gert helped me back in the early days, and Jacob helped me. But I couldn't figure out how to do it in analog.

So, I went -- you know, what people usually do, you go back to digital silicon. But it's nice 'cause it's easy to fabricate. There's you know, and hundreds of billions of dollars went into the fabs to make these things. It's fast 'cause its digital. It's deterministic, which programmers like because debugging this, you know, for most people an important part of their job function. And powering costs generally -- roughly scale with area. So, 100X smaller could be -- could be good on power and potentially cost. But you can't have anything around that arithmetic unit. You

can't have a GPU's worth of control logic, because then its pointless to make the arithmetic unit be little.

So, I looked for the simplest possible hardware design I could find, and I took it as another -- related go as you know, don't fight physics. Distance is energy. And in the mathematized machines that we all grew up with, there is no distance. Just access it in memory, it's there, grab it. Okay?

And some people worry about cash, but most of us don't try to figure out how the cashes work.

So, I wanted to expose that up to the programmers because its really important if you want to get the compute power that -- that the silicon inherently has. You have to kind of face some reality.

So, the architecture I took was just the old 80s mesh connected computers where you have a -- and this model looks like a SIMD model; it's -- the truth is, the machine does MIMD, you know, every core can do different things. But we're going to talk about it as if its every core doing the same thing.

And forgive me, one thing I do -- I've been in this thing for so long that -- like you know, I know what a SIMD -- massively connected SIMD machine is and how it works and all, but not everybody does. So,

through your training iterations, your epochs which are very slow, the error falls down something like this. But the interesting thing, of course, is that this scale is epochs, not time or energy. And if you change it to energy or time, the curve really looks like that. And for the people that are doing -- spending a month, they come in in the morning with an idea about a network they want to -- us to trick. They start it, it takes three weeks before they've got their training results, and obviously they've forgotten largely what they did and don't care about it anymore anyway -- it's way too long. So, we need a lot more compute -- at least in this deep learning field.

And the same thing works going forward. So, you know, in an embedded system, a cell phone, or an ear ring, or car, you can run these deep networks efficiently.

Okay. So, briefly the hardware and then the future and then I'll be done.

So, we built these chips, and they work. Here's one -- I brought it actually -- works. They have 2,000 cores in them. They're about a half a centimeter on a side. That's very small -- relatively small for a chip. A GPU is maybe twenty-five times

more area than this chip. This is done in a 40 nanometer silicon, which is a few generations old now. You know, modern laptops will have 14 nanometer; that's eight times denser silicon. These things run at about a couple hundred megahertz. They get about 200 gigaflops per watt peak, which is a few times better than the most recent GPUs that have come out. But they're built using modern processes and this is built using an older one.

We're putting sixteen of these chips on a board. We're hooking it to a Lynx system, or an ARM. We're building five of these for DARPA, so that's 170,000 core system. It produces about 68 teraflops peak. More interesting is it has about 68 terabytes per second memory bandwidth, which is a lot higher than the corresponding GPU system. Because they're going out currently over a bust to a separate memory. And any time you -- you know, it's the opposite of (indiscernible) local computing, and it kills -- it kills us.

If you want to get the real power out of silicon, I don't think you can use that kind of architecture.

This -- this system's going on the net

probably this summer for people to explore. There are some folks who have first dibs on it, but DARPA's being very generous and saying anybody that has an interesting project is welcome to try and use it.

And we also have these little embedded systems that are available for certain groups.

This stuff -- the commercial people out here -- this stuff is in patents.

So, there's a, you know, image of the chip. My company -- another company in Boston -- Cadence, did our physical design. MOSIS did -- and PW Run and Global Foundries did the actual silicon.

Here's a modern GPU. That's a current high-end GPU, desktop GPU done in 28 nanometer. Here's this chip we built: 3,000 cores -- 2,000 cores. That one has -- runs faster, but it's a hundred times the power driving that machine.

And so the final slide here is we need to continue building up our tools and libraries. I'm working on a deep learning library to make it easier for people in that field to try things. Building up the community of users starting with the ones I showed you in the prior slide. People doing real applications this year I hope.

C E R T I F I C A T I O N

I, MARY INDOMENICO, AN APPROVED COURT TRANSCRIBER, DO HEREBY CERTIFY THAT THE FOREGOING IS A TRUE AND ACCURATE TRANSCRIPT FROM THE AUDIO RECORDING PROVIDED TO ME BY THE OFFICE OF WOLF, GREENFIELD & SACKS, P.C., IN THE PROCEEDINGS IN THE ABOVE ENTITLED MATTER.

I, MARY INDOMENICO, FURTHER CERTIFY THAT THE FOREGOING IS IN COMPLIANCE WITH THE ADMINISTRATIVE OFFICE OF THE TRIAL COURT DIRECTIVE ON TRANSCRIPT FORMAT.

I, MARY INDOMENICO, FURTHER CERTIFY THAT I NEITHER AM COUNSEL FOR, RELATED TO, NOR EMPLOYED BY ANY OF THE PARTIES TO THE ACTION IN WHICH THIS HEARING WAS TAKEN, AND FURTHER THAT I AM NOT FINANCIALLY NOR OTHERWISE INTERESTED IN THE OUTCOME OF THE ACTION.

Mary C. Indomenico

Mary C. Indomenico

September 8, 2020

212 Vineland Avenue, East Longmeadow, MA 01028

413-746-1778

perfectinprint@aol.com